Advances in Automated Generation of Convolutional Neural Networks from Synthetic Data in Industrial Environments

Jan Hodapp
Daimler AG
jan.hodapp@daimler.com

Markus Schiemann
Daimler AG
markusschiemann@daimler.de

Vadym Bilous
BTU Cottbus-Senftenberg
bilous@b-tu.de

Claudio Salvatore Arcidiacono
Daimler AG
claudiosalvatore.arcidiacono@mail.polimi.it

Matthias Reichenbach
Daimler AG
matthias.reichenbach@daimler.com

Abstract

The usage of convolutional neural networks has revolutionized data processing and its application in the industry during the last few years. Especially detection in images, a historically hard task to automate is now available on every smart phone. Nonetheless, this technology has not yet spread in the industry of car production, where lots of visual tests and quality checks are still performed manually.
Even though the vision capabilities convolutional neural networks can give machines are already respectable, they still need well prepared training data that is costly and time-consuming to produce. This paper describes our effort to test and improve a system to automatically synthesize training images. This existing system renders computer aided design models into scenes and out of that produces realistic images and corresponding labels. Two new models, Single Shot Detector and RetinaNet are retrained under the use of distractors and then tested against each other. The better performing RetinaNet is then tested for performance under training with a variety of datasets from different domains in order to observe the models strength and weakness under domain shifts. These domains are real photographs, rendered models and images of objects cut and pasted into different backgrounds. The results show that the model trained with a mixture of all domains performs best.

1. Introduction

Modern automation and robotic support to manual labor has long struggled with the incompetence of machines to emulate human senses, especially the incredible capabilities for detection, classification and localization of human vision. Classical methods like linear classifiers or template matching have not been able to stand up to the complexity handled by our brains to analyze the information coming from our eyes.
The rise of convolutional neural networks (CNNs) since the ImageNet 2012 competition [1] shows the potential to come a lot closer to the solution of this problem. Since then, a wide variety of neural network models specialized for classification, detection (localization in the image)[2] and even six dimensional pose estimation [3] from 2D images have been proposed.
The uses of these networks in industrial environments are diverse and target areas that are still either very costly or impossible to automate: Quality checks and screenings are still mostly manual tasks, it is conceivable that image classification with neural networks can completely automate this area. Also robot part handling during production still relies mostly on predefined, precise part positioning. With localization and pose estimation robots could be able to perform complicated assemblies out of bulk goods. In this way the automation of very small batch numbers becomes feasible. Another important and still completely manual task is the assessment of risk situations inside the factory. Safety issues play an important role in human-robot collaboration (HRC), especially in Germany. Due to the numerous safety guidelines [4], among others, the main focus is on the evaluation of the different risks that can occur in the interaction between humans and robots. A system that could simplify the risk assessment process would be a significant advance, especially for HRC systems in a production environment.

The bigger picture of this paper is the application of small collaboration-robots, so called assistants, that are supposed to be configured and programmed by a blue collar worker. One skill the assistant needs is to detect and localize objects with a camera, see figure 1. To achieve this kind of usability all controls and configurations of the assistant must be easy and intuitive, so the idea is to find a way to automate the

URI: https://hdl.handle.net/10125/64307
978-0-9981331-3-3

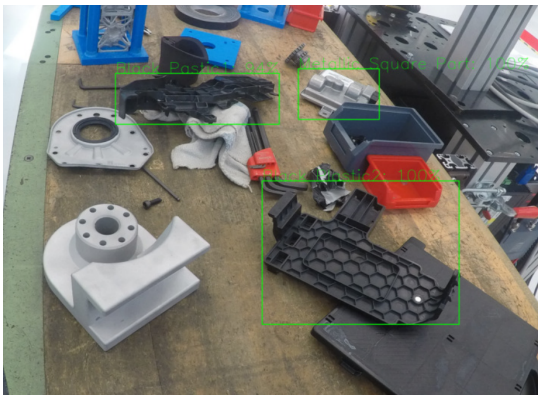

Figure 1. Example for the detector performing on objects in a cluttered scene. Figure taken from [5].

training process of a vision detection system in such a way that it becomes usable for the worker. This becomes possible with the development of transfer learning, taking a pretrained object detection model and retrain it for a new object. The biggest remaining challenge is how good training data of industrial parts can be acquired, since there rarely even exist many images of these parts, let alone labelled ones.

A solution proposed earlier this year is the usage of data from computer aided design (CAD) to generate this training data [6] by rendering it with computer graphical software (CGS).

This paper lays out two methods to improve the detector. The first method is choosing other pretrained models as core of the detector and testing them against each other. The model used in [6] was Faster R-CNN [7], the new candidates are Single Shot Detector (SSD) [8] and RetinaNet [9], both single stage detectors that promise similar performances for smaller computational cost.

The second method takes a look at how the training data are produced. While [6] just uses the CGS to render CAD data to test for viability, this paper explores the performance under real data, which was manually annotated, CAD data and a new method called cut-paste, in which pictures of the objects are taken in front of a green screen and pasted into other images.

Also the concept of distractions is introduced. This means manipulating the synthetic training data in such a way that the neural network has to learn from more difficult data, e.g. having objects in the training images that are very similar to the desired objects or that the network had problems differentiating before.

2. Method

A previous paper [6] explains en détail our method of producing training data for neural networks automatically out of CAD data in industrial environments. In short, CGS was used to realistically render 3D models of industrial parts into photos while randomizing perspective, lighting and background. These images were then used to retrain a object detection model, Faster R-CNN pretrained on the COCO dataset [7] via transfer learning. Even though the selection of an appropriate network model and training data composition was naive for this first trial, it showed promising results. The underlying method has been improved in several ways and more rigorous tests have been set in place to reliably assess the performance of the detectors. The first improvement is choosing two newer neural networks and testing them against each other. The second improvement takes the winner and tests a variation of training data compositions from different sources for performance.

2.1. Choice of Network

In the earlier trials Faster R-CNN has been used for a first, quick test, mainly because of it's good precision vs. it's computational costs. Since then a new class of detectors, one stage detectors, have entered emerged, out of these Single Shot Detector (SSD) [8] and RetinaNet [9] are tested in this paper.

In contrast to two stage detectors, e.g. Faster R-CNN, SSD eliminates the generation of region proposals by generating detection scores for default boxes, adjusting these boxes and combining scores of multiple feature maps at different resolutions in order to generalize over variable object sizes.

RetinaNet, also a one stage detector, improves on the Resnet-FPN [10] network architecture mainly by implementing a new loss function called "focal loss". Focal loss is proposed as a solution to the imbalance of numbers of examples for positive class and negative class when differentiating between foreground and background. Previously the strategy for dealing with this was hard example mining, meaning that only the errors of predictions with a loss value above a given threshold are backpropagated. Lin et al. solve this problem by re-weighting the error of examples depending on whether their classification was easy or hard. The loss function reads as follows:

$$FL(p_t) = -1(1-p_t)^{\gamma}\log(p_t)$$

from [9], with γ being a hyperparameter ranging from 0 to 5 depending on how strongly the focal loss effect is supposed to be and

$$p_t = \begin{cases} p, & \text{if } y \geq 1 \\ 1-p, & \text{otherwise} \end{cases}$$

p here is the predicted probability of one box containing the object and y is 1 when the box is containing the object and -1 otherwise, as explained in [9]. Lin et al. have shown this to perform better than hard example mining.

2.2. Domain Adaptation through Training Data Composition

The second improvement examined was how the training data is composed. This is quite important to test, whether the network can still reliably work under the domain shift from computer generated images to real camera images. This domain shift can have strong effects on the performance of a neural network as seen in [11]. Mainly three different kinds of training data were looked at:

CAD model based image generation A computer graphic software (CGS) is used to render 3D models of the objects to be detected, which in turn were made by computer aided design (CAD). This way a scene can be set up in such a way that many real world uncertainties like object orientation, perspective and lighting can be varied without much effort and also be assigned as labels to the appropriate image. A high level of realism can be achieved as has been shown by [12]. Similar to [13], the objects are rendered in front of a real photograph to avoid complicated environment setups, as the underlying goal is to provide the ability to automate the process as far as possible. Materials and surfaces of the objects were generated by hand as they usually aren't attached to the CAD model. As for the CGS used, [14] has shown that out of the two most popular ones, Blender and Unity 3D, Blender performs better, so this solution was studied. Apart from providing the surface texture this image generation can be completely automated, including labelling.

Cut-paste based image generation This second synthetic technique works by extracting objects from their background and pasting them into other images. The process highly depends on the quality of the object extraction from the source image. One could use images already annotated for segmentation, so that the segmentation labels already provide the information necessary, as proposed in [15]. Other solutions are using heuristics [16] or neural networks for extraction [17]. Because there are usually no big sets of image data for parts in industrial production environment the pragmatic green screen approach was used. The part was photographed from multiple perspectives in front of a green background, which can be separated from the object easily by applying a color mask, then the object can be pasted in another image easily. This method avoids the work to generate surfaces texture and CGS scene setup, but some real photographing has to be done by hand. The part looks very realistic but the incorporation into the new image lacks environment adjustments like lighting and reflections compared to the CAD method.

Real data generation This is classical approach of generating training data. Photographs of the objects in a variety of environments are taken and labels plus bounding boxes are annotated by hand.

2.3. Mean Average Precision

The tool of choice to judge the quality of an object detector is the Mean Average Precision (MAP) established in [18]. The calculation of a MAP goes as follows:
A trained network is tested with part of the real data, the test dataset. For each prediction by the network an intersection over union (IoU) with the ground truth is calculated and at IoU greater than a chosen value it is considered a valid prediction. Then precision (Pre) and recall (Rec) are calculated, as presented in [19]:

$$\mathrm{Pre} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

$$\mathrm{Rec} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

with TP: True positives, FP: False positives and FN: False negatives. Out of these the precision recall curve can be calculated, plotting Pre against Rec for scored confidence above thresholds ranging from 0 to 1, an example can be seen in figure 2. To gain an average precision AP for one class the mean of 11 precision scores associated to 11 equally spaced values is calculated, as taken from [19]:

$$\mathrm{AP} = \frac{1}{11} \sum_{\mathrm{r} \in (1, 0.1 \ldots 1)} \mathrm{p}_{\mathrm{interp}}(\mathrm{r})$$

$$p_{\mathrm{interp}}(r) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r})$$

with $p(\tilde{r})$ as the precision value associated to the recall value $\tilde{r}$.
The mean average precision is the average precision averaged over all classes at a certain IoU value, written $\mathrm{MAP}_{@\mathrm{IoU}}$

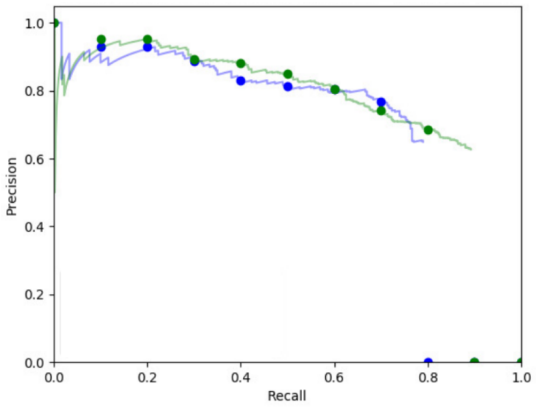

Figure 2. Two Precision-Recall curves, points at equally spaced recall values from 0 to 1. Figure taken from [5]

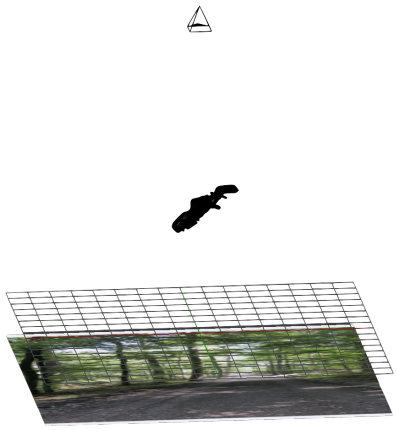

Figure 3. Blender scene setup. Figure taken from [5]

3. Implementation

This section will first describe how exactly the training data was generated by both methods. Then the setup of the two test will be discussed, namely which model is suited best for our needs and which training data composition delivers the best results.

3.1. CAD Method

Similar to [6] a scene in Blender version 2.79b was setup as visible in figure 3. The object is placed between a flat background plane and the virtual camera, the object is rotated randomly. The background of the scene, the "world texture", here in dark grey, is also textured with the background image, even though it is not visible by the camera directly, but it produces ambient lighting and reflection. The taken pictures have a resolution of 640x640. To achieve good domain adaptation several randomizations were used:

- Number of objects per image
- Kind of objects
- Position and scale of objects
- Background image and world texture
- Ambient lighting

The possible background images have been taken from the Open Images Dataset V4 [20], filtered for sufficiently high resolution.

3.2. Cut-Paste Method

Generating images over the cut-paste method involves two steps. First the object has to be photographed in front of a green screen, setup is seen in figure 4. To imitate the behaviour of the CAD method

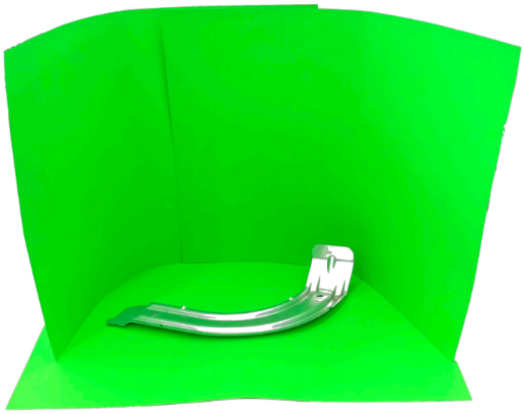

Figure 4. Green screen setup. Figure taken from [5]

videos of all possible perspectives of the object are taken and single pictures are extracted out of the video.
The second step is generating an alpha mask or transparency mask to extract the object. Because some of the objects are highly reflective but all objects are grey, the method to calculate the mask was altered from the one found in literature: [21]. Transparency α is calculated out of the RGB values:

$$\alpha(r,g,b) = 1 - \frac{g - \frac{r+b}{2}}{c}$$

This way reflective parts of the object also become partly transparent, which is very similar to the effect of reflectivity, as long as the out of view surroundings can be assumed to be similar to the background image,

which for industrial conditions holds true. The whole process is visualized in figure 5. Domain adaption is

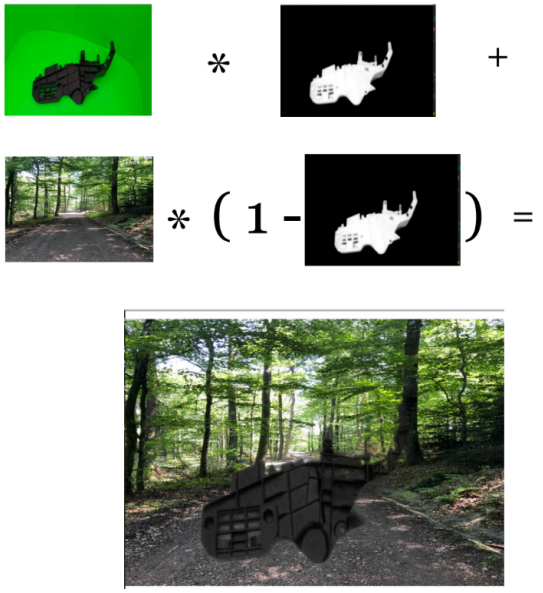

Figure 5. Cut-and-paste method. Figure taken from [5]

achieved the same way as in the CAD method with the exception of lighting, which stays constant.
For both methods there is the possibility of embedding distractions for the learning networks. Background distractions are images out of the Open Images Dataset in which false positives were detected with a higher confidence than 65%. When background distractions are turned on, all background images are composed of only this subset. Foreground distractions are other, random objects that are placed into the picture just like the desired object itself.
For comparison and real world assessment of the networks performance also 1000 real pictures were collected and manually annotated.
The objects used are up to 10 different car parts, a few examples can be seen in figure 6.

3.3. Experiments

Two experiments were made in order to determine which improvements to the automatic data generation and detection system could be made.

Model selection To select the appropriate model out of SSD and RetinaNet 6 datasets were generated, the models were trained and then tested against real images, while the datasets differentiated as follows: For each method, CAD and cut-paste (C-P), training data

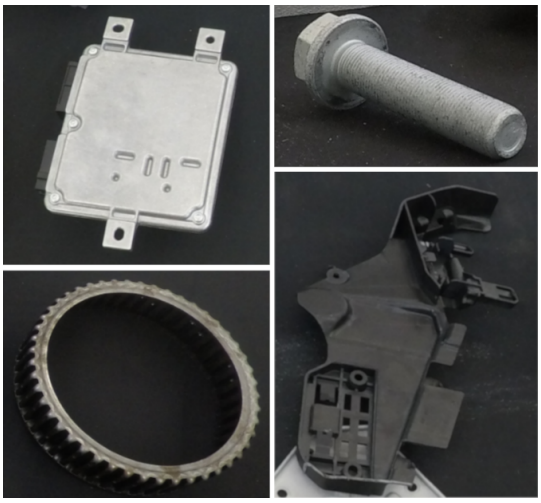

Figure 6. 4 parts on which the detectors were trained. UL: Metallic square part, UR: Screw, LL: Inner disk carrier (IDC), LR: Black plastic part.

was generated with no distractions (ND), background distractions (BD) and background plus foreground distractions (BFD). Data generation and training was done with 10 different objects from varying materials (plastic, metal, steel) and different surfaces (shiny and dull, metallic and black). 20000 training images per dataset were generated.

Training data composition For the training data composition the following datasets were generated with 6 different objects:

- 500 real images
- 10000 images with CAD method
- 10000 images with cut-paste method
- 10000 synthetic images, 5000 from CAD, 5000 from cut-paste method
- CAD plus real: 10000 images from CAD method, 500 real images
- Cut-paste plus real: 10000 images from cut-paste method, 500 real images
- Synthetic plus real: 5000 from CAD, 5000 from cut-paste method and 500 real images

The ratio of using 20 times more synthetic images than real images has proven to be successful in [22].

4. Results

4.1. Model Selection

As for the model selection, the $MAP_{@0.5}$ and $MAP_{@0.75}$ are calculated for both networks and each dataset. RetinaNet performs better in most categories

	SSD		RetinaNet	
	$MAP_{@0.5}$	$MAP_{@0.75}$	$MAP_{@0.5}$	$MAP_{@0.75}$
CAD + ND	0.60	0.48	0.68	0.59
CAD + BD	0.53	0.42	0.65	0.56
CAD + BFD	0.59	0.48	0.70	0.61
C-P + ND	0.39	0.30	0.36	0.30
C-P + BD	0.45	0.34	0.50	0.39
C-P + BFD	0.43	0.34	0.40	0.30

Table 1. Mean average precision with IoU at 0.5 and 0.75 for all model and dataset combinations.

with the exception of cut-paste method combined with both distractions and without any distractions, where it performs slightly worse, still the overall impression favours RetinaNet. Also notable is that the selection of distractions does not seem to have a consistent effect on the performance of the networks. Between each other the networks behave similarly, both showing the ranking of C-P + ND being the worst, then C-P + BFD and C-P + BD, then CAD + BD, and CAD + BFD and CAD + ND being very close first places. But for the cut-paste method only background distractions perform best, while for the CAD method the other two datasets perform better than only background distractions.

4.2. Training Data Composition

Training data composition splits into two categories, pure datasets like real, CAD, cut-paste and synthetic and real data augmented with synthetic data, which is all combinations of real and synthetic data. The model that was retrained is always RetinaNet, the test set are 500 real images. In table 2 the average precision scores at IoU of 0.5 of all six objects are shown. Table 3

	Gear	IDC	Black Plastic 1	Black Plastic 2	Bolt	Metallic Square Part
Real	0.93	0.93	0.83	0.74	0.83	0.84
CAD	0.88	0.88	0.72	0.69	0.77	0.70
C-P	0.62	0.71	0.39	0.27	0.52	0.45
Synth.	0.89	0.89	0.76	0.69	0.79	0.77
Real+CAD	0.90	0.99	0.89	0.89	0.87	0.91
Real+C-P	0.98	0.98	0.89	0.88	0.80	0.88
Real+Synth.	1.00	1.00	0.82	0.91	0.81	0.91

Table 2. Objects average precision for RetinaNet trained on all datasets.

shows the mean average precision at different IoU, the ranking of the networks performance is constant at all calculated IoU values. Notable is first of all: None of the synthetically generated datasets can compete with real images. Cut-paste performs worst by far, CAD

	$MAP_{@0.5}$	$MAP_{@0.75}$	$MAP_{@0.5...0.95}$
Real	0.85	0.77	0.70
CAD	0.77	0.66	0.61
C-P	0.49	0.41	0.38
Synth.	0.80	0.71	0.64
Real+CAD	0.90	0.81	0.75
Real+C-P	0.90	0.81	0.75
Real+Synth.	0.91	083	0.76

Table 3. Mean average precision with IoU at 0.5, 0.75 and averaged 0.5-0.95 for RetinaNet trained on all datasets.

can hold its place not too far from real and combined into "synthetic" come pretty close to the real images' performance. The advantage through the domain change between cut-paste and CAD is clearly visible.

The composed datasets, in which real images are combined with synthetic ones, beat real images every time by over 5%.

While it is hard to weigh the high expenditure of real images against its better performance compared to the low cost synthetic images, which in turn perform worse, there is no question about the advantage of using cheap synthetic data to introduce data augmentation into real training data.

5. Conclusion

For an object detector and its automatic training setup published in [6] several improvements have been introduced and tested.

First two new single shot detectors, SSD and RetinaNet, were trained with a variety of synthetic training data, which were generated automatically by either CAD rendering into realistic scenes or by cutting and pasting real photographs of the objects into real images. Overall RetinaNet showed superior performance at comparable computational cost. Therefore RetinaNet was selected as the detectors neural network.

The introduction of distractions into the training data did now show any conclusive results. The performance of both networks ranked the datasets the same way, which points at the results not being random, but with the CAD method none or all distractions performed better, while with the cut-paste method the background distractions worked better. Because RetinaNet was chosen, which showed best results under the use of background and foreground distractions this method will be used going forward.

Further improvements include enhancing the performance of synthetic data to reduce the need for real data. This could happen either by making adjustments to the synthetic data itself or optimizing the composition towards as few real images as possible while still getting good performance through domain

mixing. Also the network performs quite differently on different objects, but on certain objects it is even better with synthetic data than with real data, so there is still a lot of potential there.

Also expanding the detector to extract distance and pose of the object has huge demand in the industry to enable full localization capabilities for robots and quality check systems.

A further interest is to extract meta information out of the detector to assess accident or danger potential in images, since risk assessments are still done almost completely manually and offer great potential for automation.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, (USA), pp. 1097–1105, Curran Associates Inc., 2012.

[2] J. Han, D. Zhang, G. Cheng, N. Liu, and D. Xu, "Advanced Deep-Learning Techniques for Salient and Category-Specific Object Detection: A Survey," *IEEE Signal Processing Magazine*, vol. 35, pp. 84–100, 2018.

[3] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes," *CoRR*, vol. abs/1711.00199, 2017.

[4] ISO, *ISO 10218-1 (2011): Robots and robotic devices - Safety requirements for industrial robots - Part 1: Robots*. Geneva, Switzerland: International Organization for Standardization, 7 2011.

[5] C. S. Arcidiacono, *An Empirical Study on Synthetic Image Generation Techniques for Object Detectors*. PhD thesis, KTH Royal Institute of Technology, 2018.

[6] M. Andulkar, J. Hodapp, R. Thorsten, M. Reichenbach, and U. Berger, "Training CNNs from Synthetic Data for Part Handling in Industrial Environments," in *CASE*, (Munich), IEEE, 2018.

[7] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *CoRR*, vol. abs/1506.01497, 2015.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," *CoRR*, vol. abs/1512.02325, 2015.

[9] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," *CoRR*, vol. abs/1708.02002, 2017.

[10] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature Pyramid Networks for Object Detection," *CoRR*, vol. abs/1612.03144, 2016.

[11] R. Gopalan, R. Li, and R. Chellapa, "Domain adaptation for object recognition: An unsupervised approach," in *2011 IEEE International Conference on Computer Vision (ICCV)*, pp. 999 – 1006, IEEE, 9 2011.

[12] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 00, pp. 3234–3243, 6 2016.

[13] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning Deep Object Detectors from 3D Models," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, (Washington, DC, USA), pp. 1278–1286, IEEE Computer Society, 2015.

[14] J. Abdul, L. Farrawell, F. Jake, and C. S. K, "Training Deep Neural Networks for Detecting Drinking Glasses Using Synthetic Images," in *Neural Information Processing* (Liu Derong, , S. Xie, and Li Yuanqing, and Zhao Dongbin, and and El-Alfy El-Sayed M, eds.), (Cham), pp. 354–363, Springer International Publishing, 2017.

[15] O. Khalil, M. E. Fathy, D. K. E. Kholy, M. E. Saban, P. Kohli, J. Shotton, and Y. Badr, "Synthetic training in object detection," in *2013 IEEE International Conference on Image Processing*, pp. 3113–3117, 9 2013.

[16] C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts," *ACM Trans. Graph.*, vol. 23, pp. 309–314, 8 2004.

[17] D. Dwibedi, I. Misra, and M. Hebert, "Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection," *CoRR*, vol. abs/1708.01642, 2017.

[18] E. Mark, L. Van Gool, Williams, and Christopher K I. and Winn John and Zisserman Andrew, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 6 2010.

[19] A. Godil, R. Bostelman, W. Shackleford, T. Hong, and M. Shneier, "Performance Metrics for Evaluating Object and Human Detection and Tracking Systems," tech. rep., NIST, 2014.

[20] "Open Images Dataset V4," 2017.

[21] Z. Wu, R. Chang, J. Ma, C. Lu, and C.-K. Tang, "Annotation-Free and One-Shot Learning for Instance Segmentation of Homogeneous Object Clusters," *CoRR*, vol. abs/1802.00383, 2018.

[22] H. A. Alhaija, S. K. Mustikovela, L. M. Mescheder, A. Geiger, and C. Rother, "Augmented Reality Meets Computer Vision : Efficient Data Generation for Urban Driving Scenes," *CoRR*, vol. abs/1708.01566, 2017.